

1 This application is submitted in the name of the following inventor:

2	3 <u>Inventor</u>	<u>Citizenship</u>	<u>Residence Address</u>
4	Cohen, Earl	United States	43750 Cameron Hills Drive
5			Fremont, CA 94539

6

7 The assignee is Cisco Technology, Inc., a California corporation having an

8 office at 170 West Tasman Drive, San Jose, California 95124.

9

10 Title of the Invention

11

12 Route/Service Processor Scalability Via Flow-Based Distribution of Traffic

13 *ensa.* →

14 Background of the Invention

15

16 1. *Field of the Invention*

17

18 The present invention relates to architecture for a router. More particularly,

19 the present invention relates to architecture for a router with scalable processing power.

20

2. *The Prior Art*

Computers generate and utilize large amounts of data. Different components in a computer network, for example, several different computers each with an internet connection or "uplink", need to be linked together to allow for the transfer of data among the different components in the network. Routers perform this function. Routers take data input from one component in a network and ensure that it is properly transferred to another component in the network.

Different components in a computer network may also generate data in different forms and at different speeds. A router system in conjunction with other network interface components can properly bundle different types of information, transmit the information among different components in a computer network, and ensure that each component of the computer network is allowed to give and receive data at a rate proper for that particular component. Examples of network interface components include Ethernet transceivers and CT3 HDLC channel link data managers (as embodied in the PMC-Sierra CT3 interface chip).

Data bytes are generally bundled into "packets", a discrete grouping of information bytes that will be transferred together along the router system. Packets containing related information in a logical order are grouped into "flows". A flow consists of a unidirectional stream of packets to be transmitted between a particular source and a

1 particular destination. Packets within the same flow will have the same  
2 source/destination address, the same source/destination port, and the same protocol. It is  
3 required to maintain the packet ordering within a flow.

4  
5 Speed and accuracy are two important features of routers. If the data trans-  
6 fer rate, or throughput rate, is not rapid enough, the end user of the required data must  
7 wait for it to arrive, wasting valuable processing time. Also, input and output queues  
8 waiting to transfer data may overflow if information is being added more rapidly than it  
9 is being removed. Accuracy in transfer is also important to ensure that data arrives at its  
10 proper destination in the proper order.

11  
12 Routers can also provide services as they perform the data transfer or  
13 switching. Some of these services involve gathering information about the data being  
14 transferred or performing some other processing function upon the data as it is being  
15 transferred from one component in the computer network to another. Some of the more  
16 recent service developments involve "touching", or performing some processing func-  
17 tion, on the data in most or every packet as it is transferred. Such "high touch" services  
18 can require the router system to have a great deal of processing power. New services are  
19 being developed all the time, and thus the amount of processing that performing services  
20 requires is unbounded.

1 Current routing systems generally operate on a “one processor per line  
2 card” model. A line card is a group of components in a computer network such as a  
3 group of modems or an internet connection which sends and receives data. In a “one  
4 processor per line card” model, all of the processing for a single line, or a group of  
5 related lines, is done by a dedicated processor. This model works acceptably well if the  
6 line’s rate of data input, or input interface rate, is low enough so that the processor does  
7 not become a bottleneck in the system. With low enough interface rates, one processor  
8 can handle multiple lines.

9  
10 However, the line’s interface rate may increase as higher-speed computers  
11 or internet connections are added. Furthermore, additional router services may become  
12 desirable. Eventually, the dedicated processor may no longer be able to keep up with the  
13 processing requirements of the line and the throughput rate will slow down. If the  
14 throughput rate becomes unacceptably slow, the only solution is generally to replace the  
15 dedicated processor with a more powerful machine. This is an expensive solution. In  
16 certain cases, for example a high-speed internet line or “uplink”, a processor may not  
17 even be available that can keep up with the interface rate of the high-speed line. Current  
18 internet uplink rates such as Gigabit Ethernet (1 gigabit per second) are already becoming  
19 difficult to handle with a single dedicated processor.

20  
21 Accordingly, it would be desirable to provide a router architecture that  
22 would be easily scalable to accommodate future router service growth as well as expand-

1 able uplink connections, and which would allow processors to be added as more proc-  
2 essing power became necessary, without requiring the entire system's future hardware  
3 needs to be purchased initially. It would also be advantageous to maintain packet order-  
4 ing within each flow while still flexibly routing among different processors as necessary.

### 5 6 Summary of the Invention

7  
8 The invention provides a system and a method for a router architecture that  
9 is scalable, that is, as more processing power is desired, more individual processors can  
10 be added. The data flow from each line can be distributed among all of the processors in  
11 the system. As desired services are added, increasing the amount of "touch" or process-  
12 ing performed on the packets in the system, more processors can be added to carry the  
13 increased load. The router architecture is also able to distribute the high interface rate of  
14 an uplink connection in the same manner.

15  
16 In a preferred embodiment, packets are allocated to processors in a manner  
17 that allows the original order of data packets within the same flow to be maintained. The  
18 system uses a hash function to distribute the flows, making sure that packets within the  
19 same flow are sent to the same processor so that the original packet order in each flow is  
20 maintained. Different flows may be sent to different processors.

## Brief Description of the Drawings

Figure 1 shows a logical block diagram of the router architecture of the present invention.

Figure 2 shows a sample flow diagram for a data packet.

Figure 3a shows an example port adapter design for an Ethernet interface.

Figure 3b shows an example port adapter design for a channelized T3 line (CT3) interface.

Figure 4 shows an example Data Compiler.

Figure 5 shows an example packet digest format.

## Detailed Description of a Preferred Embodiment

In the following description, a preferred embodiment of the invention is described with regard to preferred process steps and data structures. However, those skilled in the art would recognize, after perusal of this application, that embodiments of the invention may be implemented using a set of general purpose computers operating

1 under program control, and that modification of a set of general purpose computers to  
2 implement the process steps and data structures described herein would not require undue  
3 invention.

4  
5 The present invention provides a router architecture capable of implement-  
6 ing a wide variety of services while balancing the router system load among a number of  
7 processors. The router architecture of the present invention is also capable of distributing  
8 the load from a single uplink among the multiple processors. The ability to distribute the  
9 router system load among all of the processors in the system makes the system through-  
10 put scalable; the throughput rate increases with each additional processor. Because there  
11 is no cooperation problem among the processors and each is acting independently and  
12 preferably on separate flows, the system throughput is expected to scale approximately  
13 linearly with the number of processors.

14  
15 The use of multiple processors which are redundant also provides added  
16 reliability. If one processor fails, the processing tasks can be allocated among the  
17 remaining processors.

18  
19 FIG. 1 shows a logical block diagram of the components in the router sys-  
20 tem. FIG. 1 is intended to show the interconnections and resulting scalability of the  
21 router system architecture, and is not meant to be an accurate representation of the num-  
22 ber of the various components in the router system in any particular embodiment. Indi-

1 vidual data interface components 10 are connected to a set of Port Adapters ("PA"s) 20,  
2 which are in turn connected to a set of Data Compilers (DCs) 30. The DCs 30, an  
3 Uplink 60, and a set of Route Processing Engines (RPEs) 50 are all interconnected  
4 through an Xbar 40. The Uplink 60 is connected to an external network such as the  
5 internet 70. Data transfer originates at either the individual data interface components 10  
6 or the internet 70 and travels through the router system to the Xbar 40, where it is sent on  
7 to be processed by a selected one of the RPEs 50. Once processing is complete, the data  
8 is transferred back through the Xbar 40 to its final destination, either the internet 70 or an  
9 individual data interface component 10.

10  
11 Data flow through the system is more easily followed through a sample  
12 flow diagram for a data packet as shown in FIG. 2. FIG. 2 shows only one example of a  
13 data packet flow path; numerous other paths are possible using a router system of the pre-  
14 sent invention. FIG. 2 shows an individual computer generating the original data. This  
15 data may be sent out via modem to be routed to another part of the router system. Meth-  
16 ods of transferring data via modem are well known in the art of computer networking  
17 systems. The example in FIG. 2 of an individual computer and a modem is purely illus-  
18 trative of an individual data interface component 10 in a router system as shown in  
19 FIG. 1. Other types of data interface components could serve the same function; for ex-  
20 ample, a T1 connection (1.5 Mbps) could also function to provide data to a PA 20.



1 In step 110, the data is first sent to the PA 20. A single PA 20 is capable of  
2 handling the throughput from multiple modems or other individual data interface compo-  
3 nents 10 since such components send data at a relatively slow and intermittent rate. The  
4 PA 20 bundles this data it has received from multiple modems or other interface lines  
5 into packets. Each packet is represented as a series of descriptors in a format compatible  
6 with the PA 20. Network interface components, such as many Ethernet transceivers and  
7 HDLC channel managers, use a descriptor approach to transfer packet data to a local  
8 memory, possibly with the help of a local microprocessor. Each descriptor effectively  
9 indicates a memory block (called a particle) and a size, as well as control information  
10 such as whether this is the first or last descriptor in a packet. These packets are the main  
11 units of data sent across computer network systems. The creation and use of data packets  
12 are well known in the art of computer networking systems.

13 Various types of PAs are available to convert between different user inter-  
14 faces and a generic PCI bus. The PCI (Peripheral Component Interconnect) bus is known  
15 in both the computer and networking industries. FIG. 3a shows a representative PA de-  
16 sign for an Ethernet interface. FIG. 3b shows a representative PA design for a channel-  
17 ized T3 line (CT3) interface. Other user interfaces such as OC-3 (a fiber-optic standard)  
18 may also be converted using a PA.  
19

20  
21 In step 112, the data packets from the PA 20 are sent to a DC 30. A DC  
22 provides an interface between the internal fabric in the router and a generic PCI bus

1 which can connect to a PA. Figure 4 shows a representative Data Compiler. The DC 30  
 2 further prepares the packets for transfer along the routing system by putting them into  
 3 packet digest form, and then forwards the packets along to the Xbar 40. A packet digest  
 4 contains both a header and a payload. The DC 30 will further “packetize” the incoming  
 5 packets by converting them from the particles to a consecutive block of memory, forming  
 6 the “payload” for a packet digest. The DC 30 will also add a packet digest header.

7  
 8 The packet digest header contains information about the packet to aid in the  
 9 prompt forwarding and processing of the packet, such as—the Xbar 40 source port num-  
 10 ber, the size of the packet digest header and payload, the destination RPE 50, the packet  
 11 priority, the protocol type, and the class of service required.

12  
 13 Figure 5 shows an example packet digest format<sup>200</sup>. In a preferred embodiment, the  
 14 packet digest includes the following fields:

15  
 16 o a CMD field<sup>202</sup> to control features of the packet digest (such as the TIMESTAMP field, as  
 17 described below), and to indicate a type of packet digest (where there is more than one  
 18 type, such as in alternative embodiments);

19  
 20 o a SRC\_PORT field<sup>204</sup> to indicate a source ID of the transmitting entity;

21  
 22 o a PKT\_OFF field<sup>205</sup> and a NEW\_PKT\_OFF field<sup>207</sup> to indicate an number of bytes of padding  
 23 between the packet digest header and the packet digest payload (where the PKT\_OFF in-

1 dicates a total size of the packet digest header plus any padding, while the  
 2 NEW\_PKT\_OFF field indicates a new value for PKT\_OFF);

3  
 4 o a PDH\_SIZE field<sup>210</sup> to indicate the size of the packet digest header;

5  
 6 o a DEST\_OFF field<sup>212</sup> to indicate an offset of the destination IP address in the packet digest  
 7 header;

8  
 9 o a "FIB Leaf" field<sup>214</sup> to indicate a pointer to the leaf in a FIB lookup m-trie for the destina-  
 10 tion IP address;

11  
 12 o a DEST\_MASK field<sup>216</sup> to indicate which Xbar ports are intended as destinations for the  
 13 packet (where a selected bit <39> of the DEST\_MASK field preferably indicates a multi-  
 14 cast packet);

15  
 16 o a QUEUE field<sup>218</sup> to indicate to which hardware output queue at the particular output  
 17 interface to direct the packet;

18  
 19 o an INTERFACE field<sup>220</sup> to indicate to which hardware output interface to direct the  
 20 packet;

21  
 22 o a TIMESTAMP field<sup>222</sup> to indicate a timestamp or other data to be written by the in-  
 23 bound Xbar interface (where a selected bit CMD <0> indicates that the TIME-  
 24 STAMP field should be written to); and

1  
2 o a set<sup>224</sup> of additional words reserved for software classification data.

3  
4 In order to determine the destination RPE 50, the DC 30 includes a classifi-  
5 cation engine coupled to the crossbar interface to determine which RPE 50 will receive  
6 each packet. Packets are not distributed randomly among RPEs, so as to maintain the  
7 original ordering of packets within the same flow. The classification engine distributes  
8 packets so that packets within the same flow will be sent to the same RPE, but packets in  
9 different flows may be sent to different RPEs and possibly reordered as determined by  
10 RPE resources.

11  
12 The classification engine analyzes the router system traffic and distributes  
13 flows to multiple processors. The distribution is partly table-driven, and is thus flexible  
14 and easily changeable, even dynamically. In a preferred embodiment, the classification  
15 uses a hash function of packet flow information to select a specific RPE 50 to handle  
16 processing for that packet flow. The hash function distributes packets evenly among the  
17 processors in response to flow information such as the source/destination address, the  
18 source/destination port, and the protocol. The hash function can operate using any in-  
19 formation that will allow for flow preservation.

1 In a preferred embodiment, the hash function performs an XOR (exclusive  
2 OR) logical function of several bytes of fields from the IP header of the packet, including  
3 the following fields:

- 4
- 5 o a source IP address (four bytes);
- 6
- 7 o a destination IP address (four bytes); and
- 8
- 9 o a protocol type value (one byte).

10  
11 The XOR logical function is performed for all nine bytes, providing a sin-  
12 gle byte output value, which can be used to index into a 256-entry table for selecting the  
13 RPE 50.

14  
15 Once the DC 30 has determined the packet's destination RPE 50 and the  
16 packet header and payload are ready to be sent out, the packet (now in packet digest  
17 form) is placed in an input queue to be stored until it is sent to the selected RPE 50. Note  
18 that the amount of processing performed by the DC 30 on each packet is bounded; the  
19 DC 30 only does a fixed amount of work per packet. There is no cost in terms of DC 30  
20 processing for adding additional services, as all such additional processing will be per-  
21 formed by the RPEs 50. Thus, the DC 30 is prevented from becoming a system bottle-  
22 neck as more services are added. In the router system of the present invention, DC 30

1 input queues should only overflow if there are not enough RPEs 50 to support the re-  
2 quested level of services at the incoming data rates; if input queues overflow, it is always  
3 possible to add more RPEs 50.

4  
5 In step 114, packets are sent from the DC 30 to the Xbar 40. The Xbar 40  
6 provides the interconnection between the data origination component ports, the router  
7 system processor ports, and the data destination component ports. In one data transfer,  
8 the data packet will cross through the Xbar 40 twice (step 114 and step 118 in FIG. 2).  
9 The Xbar 40 receives an incoming packet and transfers it to the selected RPE, and then  
10 the Xbar 40 transfers the packet back to its destination port. The Xbar 40 must provide  
11 enough bandwidth to handle the data interface rates of both the DCs 30 and the  
12 Uplink 60. Xbar interconnections are known in the art of computer and router systems.

13  
14 The Xbar 40 includes a number of ports; each port is a pair of uni-  
15 directional links. The Xbar 40 implements a simple arbitration scheme between sending  
16 and receiving components, ignoring whether the destination port has the resources to  
17 handle the packet or not. Packets can be dropped at any point if there are not sufficient  
18 resources to handle them. Software must ensure that there are sufficient resources avail-  
19 able at a destination port, for example, memory space at the output to queue the packet,  
20 before sending a packet over the Xbar 40. Output queuing protocols determine what oc-  
21 curs when an output queue overflows; output queue management is performed at the  
22 DC 30.

1  
2 The Xbar 40 includes an arbiter — ports forward their requests to the arbi-  
3 ter, and grants are sent back. The Xbar 40 arbiter uses a windowing scheme in which  
4 each port can make requests for multiple destinations at the same time; the arbiter at-  
5 tempts to find the best fit between requesting ports and destinations. The Xbar 40 can  
6 also support multicast — a port can request that a packet be sent to multiple destinations.  
7 In the preferred embodiment, the Xbar 40 waits for all ports to be free, and then sends the  
8 multicast request all at once. Numerous other Xbar multicast schemes are well known in  
9 the art.

10  
11 The Xbar 40 in the preferred embodiment discussed herein is packet-based,  
12 but the Xbar 40 could be designed to be cell-based and still fall within the inventive con-  
13 cepts disclosed herein. It is also possible to provide multiple Xbars or a bus that per-  
14 forms the same function as the Xbar. The methods discussed herein for the implementa-  
15 tion of such an Xbar device are by no means limiting. Alternate methods for the imple-  
16 mentation of such an Xbar device will readily suggest themselves to those of ordinary  
17 skill in the art.

18  
19 In step 116, the Xbar 40 transfers the packet to the destination RPE 50.  
20 There are multiple RPEs 50 connected to the Xbar 40 that provide both switching and  
21 services, including the high-touch type of services that may require a great deal of proc-  
22 essing power. Such services may also be added later to the system as more services are

1 developed and become available. Additional RPEs 50 can be added to the router system  
2 to provide additional processing power; this provides the scalability feature of the present  
3 invention.

4  
5 An RPE 50 receives a packet digest with a packet digest header containing  
6 classification information provided by the DC 30. Based on the classification informa-  
7 tion, the packet is put into one of a small number of input queues on the RPE 50. The  
8 intent of the RPE 50 queuing is to provide the capability for handling both high- and low-  
9 priority packets. Packets are processed from among the RPE's input queues by the  
10 RPE 50 processor in whatever order it sees fit.

11  
12 The RPE 50 processor will perform various tasks on the packet, including  
13 switching, tag application/update, access list processing (filtering), and all other router  
14 service functions. Some of these operations result in the RPE 50 modifying the packet  
15 digest, others only update RPE-internal data structures. When the RPE 50 processor is  
16 done with a packet, the RPE 50 will know the packet's final destination. The example  
17 destination in FIG. 2 is the internet 70 via the Uplink connection 60. However, it would  
18 be evident to one of ordinary skill in the art that the packet's destination could also be an  
19 individual data interface component 10 via a DC 30 and a PA 20.

20  
21 In step 118, the packet is put back on the Xbar 40 from the RPE 50. In  
22 step 120, the Xbar 40 transmits the packet to the Uplink 60. The Uplink 60 is a high-



1 bandwidth PA with its own DC. Like a DC, the Uplink 60 is connected to the Xbar 40  
2 via an Xbar link. However, in order to meet its bandwidth requirements, an Uplink 60  
3 may require multiple Xbar ports. The Xbar 40 can support multiple Uplink 60  
4 connections.

5  
6 In step 122, the Uplink 60 transfers the data to the internet 70, completing  
7 the data transfer from the individual data interface component 10 to the internet 70.

8  
9 The Uplink(s) 60 can also send data to the Xbar 40 for transmission to an  
10 RPE 50 and eventual transfer to an individual data interface component 10. The  
11 Uplink(s) 60's traffic from the internet 70 is also distributed via the Xbar 40 to the vari-  
12 ous RPEs 50 using a hash function. In this way, the router system can support a plurality  
13 of Uplink(s), each of arbitrary speed.

14  
15 Various alternate embodiments of the present invention have also been  
16 conceived, and would be clear to those skilled in the art after reviewing this application.  
17 For example, in alternative embodiments, the DC and a specific PA type can be inte-  
18 grated together to produce a more cost-effective, though less easily adaptable, interface.

19  
20 Other alternative embodiments include performing some services of a  
21 bounded nature on the DCs, as is currently done with output queuing. For example, the  
22 DCs could accumulate statistics and do flow-based accounting on each packet. In this

1 way, the DCs could handle some portion of the known processing load leaving un-  
2 bounded and future services to the RPEs.

3  
4 Yet other alternative embodiments include adding specialized processing  
5 engines connected to the Xbar. Packets can be sent through these specialized engines  
6 either before or after or instead of an RPE to perform services such as compression/  
7 decompression, encryption, or routing. The classification engine on a DC could  
8 determine the type of service required by a packet and route it appropriately.

9  
10 While embodiments and applications of this invention have been shown  
11 and described, it would be apparent to those skilled in the art that many more modifica-  
12 tions than those mentioned above are possible without departing from the inventive con-  
13 cepts herein. The invention, therefore, is not to be restricted except in the spirit of the  
14 appended claims.

1 91. The method as in claim 90 further comprising:

2 processing compression as said type of service.

1 92. The router as in claim 90, further comprising:

2 processing decompression as said type of service.

1 93. The router as in claim 90, further comprising:

2 processing encryption as said type of service.

1 94. The router as in claim 90, further comprising:

2 processing routing as said type of service.

1 95. A computer readable media, comprising:

2 said computer readable media containing instructions for execution in a processor

3 for the practice of the method of claim 17 or claim 71 or claim 90.

1 96. Electromagnetic signals propagating on a computer network, comprising:

2 said electromagnetic signals carrying instructions for execution on a processor for

3 the practice of the method of claim 17 or claim 71 or claim 90.